AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions of claims in the application:

Listing of Claims:

1. (Currently Amended) A system embodied on a computer-readable storage medium that facilitates extending the functionality of an application, comprising:

a schema component that includes a schema element representative of domain terminology of a problem for solving ~~to be solved in~~ a host application, the domain terminology is ~~not~~ different from native terminology utilized in ~~to~~ a general application programming interface (API) of the host application; ~~and~~

a mapping component that maps the schema element to a construct of an API of the host application such that domain terminology maps to native terminology and enables the host application to operate on the domain terminology; and

a generation component that produces a new API based upon the mapping, wherein a host application facilitates document development using domain terminology related to the problem.

2. (Previously Presented) The system of claim 1, the host application is at least one of a word processing application, spreadsheet application, drawing application, presentation graphics application, website design and development application, database application, project management application, publication application, note management application or browser and communication application.

3. (Original) The system of claim 1, the schema component facilitates generation of at least one of a data programming model and a view programming model.

4. (Currently Amended) The system of claim 1, ~~further comprising a~~the generation component that generates at least one of a data programming model and a view programming model.

5. (Previously Presented) The system of claim 1, further comprising a generation component that generates a data programming model and a view programming model that are automatically connected to each other *via* data binding, the view programming model provides an interface by which the host application operates on the domain terminology.

6. (Original) The system of claim 1, further comprising a generation component that generates at least one of a data programming model and a view programming model, wherein the data programming model interfaces to the host API *via* the view programming model.

7. (Original) The system of claim 1, further comprising a separation component that separates data from document content.

8. (Currently Amended) The system of claim 1, further comprising a separation component that generates a data island in a document of ~~a~~ the host application.

9. (Original) The system of claim 8, the data island is editable.

10. (Previously Presented) The system of claim 8, the data island can be at least one of accessed or modified without launching the host application.

11. (Original) The system of claim 8, contents of the data island and contents of the document are synchronized when the document is run inside the host application *via* data binding.

12. (Currently Amended) The system of claim 1, the schema component and the mapping component facilitate generation of a new API that interfaces to the ~~host~~ API of the host application and enables the host application to operate on the domain terminology.

13. (Cancelled).

14. (Currently Amended) A computer, comprising:

a schema component that includes a schema element representative of domain terminology of a problem for solving ~~to be solved in~~ a host application stored ~~at the~~ in a storage media, the domain terminology is ~~not part of a~~ different from native terminology utilized in an application programming interface (API) ~~API~~ of the host application; ~~and~~

a mapping component that maps the schema element to a construct of an API of the host application and enables the host application to operate on the domain terminology;

a generation component that produces a new API based upon the mapping, wherein a host application facilitates document development using domain terminology related to the problem; and

processing hardware to execute at least one instruction~~instructions~~ associated with the schema component[[and]], mapping component, or generation component.

15. (Previously Presented) The system of claim 1, the schema component includes a view schema that represents only data of interest of the host application.

16. (Previously Presented) The system of claim 1, the schema component includes a view schema that facilitates pulling a plurality of objects of interest from a plurality of APIs of the host application.

17. (Previously Presented) The system of claim 1, at least one of the schema component and the mapping component facilitate generation of a view API that is a hybrid of view schema and the API of the host application.

18. (Currently Amended) A system embodied on computer-readable storage media that facilitates extending the functionality of an application, comprising:

a schema component that includes a schema in terms of a problem ~~to be solved in~~ for solving a host application that ~~are not~~ is different from native terminology utilized in ~~to~~ the host application and a mapping of the terms to generic objects of an API of the host application; and

a generation component that generates ~~at least one programming model based on the schema that interfaces to the API~~ a new API based upon mapping of domain terminology to native terminology, wherein a host application facilitates document development using domain terminology related to the problem.

19. (Original) The system of claim 18, further comprising a separation component that generates an editable data island in a document of the host application.

20. (Original) The system of claim 19, contents of the data island and contents of the document are synchronized when the document is run inside the host application.

21. (Currently Amended) The system of claim 18, the schema component facilitates generation of a new API that interfaces to the API of the host application.

22. (Original) The system of claim 18, the schema component facilitates manipulation of a variable without reference to underlying register and stack allocations.

23. (Previously Presented) The system of claim 18, further comprising a separation component that generates an editable data island in a document of the host application, which data island is accessible and modifiable without running the host application.

24. (Currently Amended) A computer implemented method of extending the functionality of an application, comprising:

creating a schema of problem domain elements of a problem ~~to be solved in~~ for solving, the domain elements are ~~not native~~ different from native elements utilized in ~~to~~ an API of the application;

mapping the problem domain elements to constructs interpretable by one or more generic application interfaces of the application; and

generating a new API based upon the mapping, wherein a host application facilitates document development using domain terminology related to the problem ~~program model based on the mapping of the problem domain elements such that the one or more generic application interfaces can be accessed *via* the program model using the problem domain elements~~.

25. (Currently Amended) The computer implemented method of claim 24, further comprising automatically separating the program model into a data model and a view model.

26. (Currently Amended) The computer implemented method of claim 24, further comprising exposing data of the problem domain elements as named objects in a view model.

27. (Currently Amended) The computer implemented method of claim 24, further comprising exposing data of the problem domain elements as declarations in a data model.

28. (Currently Amended) The computer implemented method of claim 24, the program model is a schema-based, machine generated model.

29. (Currently Amended) The computer implemented method of claim 24, further comprising exposing data of the problem domain elements as first class named objects.

30. (Currently Amended) The computer implemented method of claim 24, further comprising separating data from a view model of the program model by,

generating data that conforms to the schema; and

saving the data as a data island in a document of the application.

31.-37. (Cancelled)

38. (New) A method for facilitating relationship building, comprising:

employing a processor executing computer executable instructions stored on a computer readable storage medium to implement the following acts:

constructing a view model;

interfacing the constructed view model with a host model through using a control mechanism;

constructing a data model; and

interfacing the data model to the host model indirectly through the view model *via* data binding to the view model.

39. (New) A method for facilitating extension of application functionality, comprising:

employing a processor executing computer executable instructions stored on a computer readable storage medium to implement the following acts:

collecting terminology of a problem domain;

determining constructs in a generic host application programming interface that corresponds to domain terminology;

mapping the determined constructs to corresponding domain terminology;

generating a new application programming interface based upon the mapping; and

enabling a host application to use the new application programming interface, the host application uses the new application programming interface for document development.